22nd International Congress on Modelling and Simulation, Hobart, Tasmania, Australia, 3 to 8 December 2017
mssanz.org.au/modsim2017

Automating the Design of Battle Rhythms

M. J. Pilling [a]

[a]*Behaviour & Control Group, Decision Sciences, Joint Operations and Analysis Division, Defence Science & Technology Group, P.O. Box 1500, Edinburgh SA 5111*
Email: michael.pilling@dst.defence.gov.au

Abstract: A Battle Rhythm is the organisational process by which a Headquarters implements the decision cycle and thus generates shared situation awareness and war fighting decisions in a timely manner. In other words, a Battle Rhythm must sequence the execution of activities within a headquarters to regulate the flow and sharing of information including the generation of analysis in support of multiple decision cycles. Battle Rhythms must be synchronised across multiple Headquarters.

The ADF have a requirement to create new Battle Rhythms from time to time in response to new operational contexts and this is currently done by hand by staff with Battle Rhythm experience. Generating such a schedule of activities is both painstaking and error prone. Without an analytical framework, it is easy for the best officer to make a choice that will lead to slower response times than necessary or critical decision deadlines being missed. Such issues may not become apparent until the Battle Rhythm has been operational for some time.

Furthermore, the final Battle Rhythm does not communicate many of the requirements it seeks to meet. This makes it difficult to later add further requirements or tasks to the Rhythm and be sure that its current features will not fail. This is particularly true if such a Rhythm has to be modified by someone other than its original author.

This paper describes work in progress on a system whereby the user does not produce a particular Battle Rhythm but instead specifies the requirements they have of it. By making outputs such as decisions, documents and intermediate assessments along with deadlines and synchronisation points explicit, the system can not only automate the generation of Battle Rhythms but also provide assurances that the operational environment will meet its deadlines provided each intermediate activity described by the Battle Rhythm is completed on time.

The system will also be able to detect some faults and omissions in Battle Rhythm specifications such as overuse of resources, missing work products and general schedule infeasibility and can give meaningful diagnostic messages in these cases. The fact that requirements are fully documented allows the original specification to be easily augmented with extra requirements, should they arise during operations, and to produce a new Battle Rhythm that meets both the new and existing requirements.

This brief paper gives an overview of a Battle Rhythm specification language, a work in progress, and some of its potential. Realising the full potential of such a system will require experiments with generating real Battle Rhythms. Since examination of extant Battle Rhythms does not currently reveal all of the actual requirements behind them, the work will need to be progressed with experienced ADF officers. Only once the adequecy of the specification language is verified, along with the utility of the system, can we sensibly tackle questions of the best user interface.

Keywords: *Battle Rhythm, scheduling, constraint optimization, workflow, graph based analysis*

1 INTRODUCTION

A Battle Rhythm is the organisational process by which a Headquarters implements the OODA loop and thus generates shared situation awareness and war fighting decisions in a timely manner. In other words, one US military definition of a Battle Rhythm is "the sequencing and execution of activities within a joint force headquarters that are regulated by the flow and sharing of information that support all decision cycles." US Joint Task Force Headquarters (2012). Another view is "The purpose of Battle Rhythm management is the maintenance of synchronous activity and process among distributed war fighters. It is most critical in rapidly evolving situations or in highly distributed operations." Duffy et al. (2004)

A Battle Rhythm is a cyclic timetable for a series of meetings, briefings and work assigned to staff including information extraction activities that filter and combine information so as to present it to commanders to enable timely war fighting decisions. The ADF runs operations under such a "Battle Rhythm", in order to accumulate situation awareness and to produce military decisions on time to effectively prosecute its operations.

Fundamentally, the Battle Rhythm must organise staff activity to:

- Deliver filtered and assessed information to commanders in time to make war fighting decisions.
- Coordinate activities within and between various headquarters.
- Allow for tasking within the rhythm for particular information collection.

The ADF runs operations under such a "Battle Rhythm". From time to time, it has a need to create new Battle Rhythms and needs to not only be able to assess whether a new rhythm will meet their operation requirements but also know it is feasible for ongoing use. Possible reasons for an new Battle Rhythm would be to vary the rate of information production, or to focus on generating different awareness or decisions. Problems with a Battle Rhythm may not be immediately apparent on inspection and may not occur until the Battle Rhythm has been operating for some time. Existing Battle Rhythms often need to be tweaked as operational requirements change and any such changes need to be tested for feasibility.

Clearly being able to test the feasibility of a Battle Rhythm with its associated staff and work assignments before use would enable to ADF to have confidence that a particular rhythm will work rather than discovering problems with it in action. In particular, it is not always obvious which deadlines will not be met when a Battle Rhythm does fail.

This paper offers a language for specifying Battle Rhythm requirements, that is machine readable. The language can be used by ADF staff to describe such requirements in a way that contains sufficient information so that mathematical tests embodied as computer programs can be applied and either find a feasible schedule for a new Battle Rhythm meeting those requirements or warn that the goals are not achievable with the current constraints. Unlike the current handraulic system, staff would not create the battle rhythm *per se*, but rather specify the requirements of what they need the Battle Rhythm to achieve as well as task dependencies, staffing and other resource constraints. In particular the language provides for tracing the generation of documents and decisions throughout the Battle Rhythm.

The advantages of such an approach include:

- Automating painstaking, error prone and dreary work (producing the Battle Rhythm) [1].
- Prediction of potential or certain failures before they occur in a live system.
- The potential to have greater capacity by getting the most out of available staff without overworking them.
- Ensuring that failure in overload conditions occurs in the least or near least critical tasks. Application of computer scheduling theory ensures that the least critical tasks, as specified by the user requirements are affected first but this assumes processors can be re tasked at a moment's notice. Clearly this is not the case for humans, and so the increased granularity prevents such an optimal guarantee. However, the system will accurately predict which task(s) will miss their deadline and with this knowledge the specification can be changed to add more staff so this doesn't happen. The possibility of automatically producing multiple views or representations of the Battle Rhythm tailored to the needs of different users.

[1]Following Bruce J. MacLennan's Automation Principle. MacLennan (1983)

- The ability to flag areas of the battle rhythm that are most sensitive resource constraints and to get an overall indication of how robust the battle rhythm is.

At times this approach may require the worker to put aside unfinished tasks to work on tasks that are apparently less critical. However it will only do so to avoid a failure to meet a more highly critical deadline at a later time. Such paradoxes of work scheduling are far from obvious, especially to someone in the middle of just trying to get things done but the mathematics behind the system takes these into account. It also requires estimates of time needed to complete each task in the worst case, and some information about tasks that were previously considered irrelevant, such as their relative criticality to the operation at hand.

As well as providing a system by which to produce Battle Rhythms more simply and reliably, the specification system provides a common nomenclenture to talk about what is needed from a Battle Rhythm and more importantly focuses the mind on what considerations need to be taken into account.

Eventually such a system may allow a far more dynamic Battle Rhythm which is adapted regularly according to operational priorities.

While previous work has focused on such things as documenting and understanding ADF processes and informatin flows using such tools as the IDEF description language Menzel and Mayer (1998), or understanding the execution of tactical battle rhythms so as to provide the most appropriate collaborative technologies for communicating information (Duffy et al., 2004); our work is intended to enable the reliable and quick generation of Battle Rhythms to meet particular ADF needs.

2 WHAT IS NEEDED IN A BATTLE RHYTHM SPECIFICATION?

The users of this system will have domain expertise, but not necessarily scheduling expertise. They will know what they want to achieve from a Battle Rhythm but will not necessarily know how to build an efficient and robust Battle Rhythm, and in particular they may not detect errors or faults in a Battle Rhythm before it plays out.

The users may also be entrenched in the current handraulic manner of building Battle Rhythms and so may not be aware of what is possible because “it has always been done this way.”

In addition some user cultural proclivities may work against concepts that naturally arise in scheduling such as prioritisation. For instance a strong work ethic of we’ll “just make it work” or “just get it done”[2] can reduce the consideration of what is a practical time to allow for each task to be completed, and which tasks should be jettisoned if they can’t all be done. While this may be considered in times of overload, they may rarely be considered *a priori*.

The best specifications are those that describe true requirements while leaving as many choices as possible undefined so as to give the maximum degrees of freedom in finding a solution. They also make requirements explicit. One feature of Battle Rhythms as they are communicated within the ADF is that they generally omit the actual requirements and details of what they produce, focusing purely on the timetable for being there to perform the work. Actual requirements for document production and decisions are not documented beyond the general name of the activity or meeting. In general, the specific purposes, inputs and outputs of anything in the Battle Rhythm are currently assumed knowledge of the participants, or hidden in standard operating proceedures. Making these requirements explicit allows them to be checked for correctness, by the system itself in terms of general deadlines and the linkages between inputs, outputs and activities. Likewise when the system displays these linkages in an easily comprehensible way, the users themselves can check for completeness and sufficiency to meet the overall intent of the system.

3 A LANGUAGE VIGNETTE

Limited space precludes a detailed exposition of the specification language and all its capabilities. Nevertheless, a few examples can give a good sense of it.

Figure 1 shows a syntax diagram giving the overall structure of a Battle Rhythm specification.

3.1 Name

Each Battle Rhythm must have a name, both for human communication, and to allow the Battle Rhythm for one site to reference activities in the Battle Rhythm of another. Foreign tasks can be referenced by using their

[2]Currently, overloads are usually overcome by adding extra staff in an ad hoc manner rather than allowing the extra time needed.

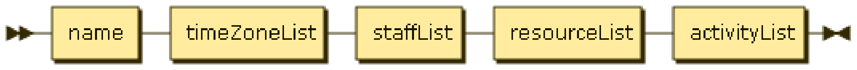

Figure 1. The basic syntactic structure of a Battle Rhythm specification.

battle rhythm name as a qualifier.

3.2 Time Zones

The time zone list provides for users to define the time zones they wish to use when elabortating their specification. The final schedule will be indexed by times in each of these time zones.

3.3 Staff and Roles

The staff list includes both a listing of staff and their roles. Each person on staff can have multiple roles. Roles can be used to specify a rank or position of authority being exercised, or they can refer to a skill or qualification possessed by that member of staff. Staff can be listed either by their name or, more generically their position if staff have not yet been assigned.

```
staff Jane_Jones, John_Smith_1, "John Smith 2", Diana_Prince;

role Commander, Targeting_Officer: Jane_Jones, "John Smith 2";
role Legal_Officer: John_Smith_1;
role Targeting_Officer: Diana_Prince;

Staff Douglas_Brown, "Anna d'Orsay";
role Legal_Officer: "Anna d'Orsay"; role targeting_officer: Douglas_Brown;

Role Day_Shift: Jane_Jones, Diana_Prince, "Anna d'Orsay";
Role Night_Shift: John_Smith_1, "John Smith 2", Douglas_Brown;
```

Figure 2. A very small staff list

Further work is required to test whether roles are an appropriate way to deal with shift work and shift interactions as substantive roles can continue across shifts but are performed by different staff. We wish to allow flexibility but require enough information to detect problems.

3.4 Declaring Resources

We also need to declare any resources we intend to use. These may be things like meeting rooms, teleconferencing equipment etc. They should all be local to the headquarters concerned. This ensures that activities that require the same resource are not scheduled simultaneously in the derived Battle Rhythm.

A resource specification is simply a list of resources giving the number of them that exist. If no number is given, the resource is assumed to be unique. The word "unlimited" is used to document resources which are ubiquitously available, or are supplied and can be shared infinitely such as printers. If a number greater than one is given, then the system will create a group of resources - in this case if the specification given is *13 meeting_room* the system would create *meeting_room_1* up to *meeting_room_13*. Similarly, if the user specified *5 "Meeting Room"* the system would create *"Meeting Room 1"* up to *"Meeting Room 5"*.

A simple example of a resource specification is given in figure 3.

3.5 Declaring Activities

Activities are the heart of a Battle Rhythm. They are high level descriptions of what the staff have to do to successfully run the headquarters. As such, declaring activities is the most variable of all the things we do in specifying a Battle Rhythm.

```
resources "Main Meeting Room", 5 "Small Meeting Room", 6 Projector,
    unlimited "A4 Printer", unlimited Telephone;
```

Figure 3. A small resource specification

An activity is a task or a meeting. They can be specified in any order and combination. The language does not deal directly with rostering which is considered a separate and subsequent activity. Meetings are only a special case of a task but are so common and fundamental to Battle Rhythms that we provide a shorthand for them that allows their specification to be less cluttered.

The options for specifying a task allow the user to declare how often a task occurs, when it first occurs, the work products (documents, data, decisions) needed to perform the task and any work products produced, and the staff and resources needed.

Importantly, the specification also allows the user to specify preparation and back briefing activities together with the task including any extra staff and resources they may need.

We will use an example of a general sync meeting in the first instance. This is the "keystone" for this Battle Rhythm, the meeting that all other activities revolve around. The meeting is held for half an hour commencing at 0700 each day.

```
task "Main Sync" occurs daily,
    starting at 07:00,
    work 00:30 estimated by George,
    due within 00:30,
    continuous,
    importance 1,
    requires "Main Meeting Room" &
             distinct Commander &
             2 "Targeting Analyst" & distinct "Legal Officer" &
             "Army_Liaison" & "Navy_Liaison" & "Air_Liaison",
    uses situtation_report & possible_target_list & vetted_target_list &
         sortie_results_report,
    produces approved_target_list for air_target_development,
    produces selected_target_list for legal_vetting,
    produces action_list & orders_list for Army_Liaison & Navy_Liaison
     & Air_Liaison,
    preparation 1:00 requires "Legal Officer" & 2 "Staff Officer",
    backbriefing 00:10 requires Army_Liaison & "General 1Div" &
                                "Meeting Room 5",
    backbriefing 00:15 requires Navy_Liaison & "Captain HMAS Sydney" &
                                "Video Conference",
    backbriefing 00:30 requires Air_ Liaison & 5 "Air Mission Planner" &
                                "Squadron Leader" & "Meeting Room";
```

Figure 4. The sync task, with support activities

Given this is the keystone task, we must also specify its start time, whereas with most tasks we would allow the system to fit them around the other tasks according to the flow of work products from one task to another.

We also need to give an estimation of how long the work will take which includes the name of the person who estimated it because task time estimation is notoriously difficult in general and feedback to the estimator is the most effective way to improve this skill. As this is a meeting, the deadline is the same as the worktime. Had we declared this as a meeting rather than a task, this would be automatically assumed as would the fact that the work had to be done in one continuous chunk.

We also wish to specify that this task is the most important of all (in the sense that the others fit in around it and feed into it). Importances are given as an integer with 1 being the most important class of tasks, 2 the second most important etc. Often only a single task is nominated as most important. It's also possible, for instance, to nominate both the morning and evening instances of a task with the same importance.

Now we state who and what resources need to be at the meeting using the *requires* clause. If this requirement omits a number for a resource or a role it is taken to be 1. Each "distinct" declaration applies only to the task it occurs in and states that the person fulfilling that role may take no others, otherwise the system is free to find a single person who can fulfil multiple roles. In this case, neither the Commander nor Legal Officer can be performing any other role in this task, but the Targeting Analysts might also be Liaison Officers.

The *uses* and *produces* clauses allow the Battle Rhythm designer to specify the work products the meeting needs to consume and produce respectively to be effective. Specifying these allows the system to infer deadlines for outputs, check that inputs are being produced and ensure they would be on time, and to check that all outputs are being utilised. The outputs clause can be repeated to specify outputs for various meetings or recipients.

It is common in Headquarters for staff officers to have to prepare or massage inputs for a meeting, and for liaison officers to have to back brief their respective units. The Battle Rhythm specification language provides special clauses to specify these activities. The *preparation* clause specifies an activity that the system will attempt to schedule immediately before it's substantive activity. Similarly the system will attempt to schedule any associated backbriefing immediatly after its associated substantive activity. Multiple instances of these *preparation* and *backbriefing* clauses can be specified for each substantive activity, and if roles are required in both an associated and the substantive activity the system will allocate the same staff to both for continuity. Preparation and debreifing activities can be scheduled simultaneously unless they conflict over resources including staff.

4 A PIPELINE OF SITUATION AWARENESS

Specifying a Battle Rhythm with explicit work products, makes it very clear that the activities create several pipelines of information that is being refined, filtered and clarified. Figure 5 shows a simple graph analysis of some of the work products used and produced across multiple activities. In this case is shows the pipeline generating targeting data and eventually targeting decisions. The arrows stand for "is based upon" or "is a prerequisite for".

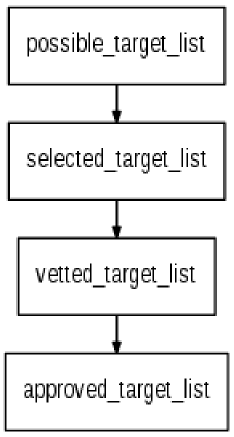

Figure 5. An information pipeline derivable from a Battle Rhythm specification

Generating such pipelines graphically for the user can assist them in ensuring the Battle Rhythm is sufficient to generate the situation awareness and decisions needed to effectively prosecute operations. Ideally, the staff officer writing the Battle Rhythm specification would start with such a hand drawn graph to inform the specification.

4.1 Keeping the pipelines full

One interesting possibility of using such a specification language to generate Battle Rhythms is to consider the special cases at the beginning and conclusion of activities. For instance, on commencement, the targeting pipeline will be empty. It may well be worth including one off activities to quickly evaluate the top two potential target, and thus "pump prime" the pipeline while the more usual wide target selection activities are begun.

With such a Battle Rhythm generation system, it is easier to try adding such extra activities for a limited period and accommodate such changes in operating phase compared to a manual Battle Rhythm production system.

5 IMPLEMENTATION

The Battle Rhythm language is general and allows the system to utilise the simplest scheduling system needed to produce a schedule that meets the deadlines and dependencies in the specification. This can include rate monotonic Liu and Layland (1973) – the optimal static scheduling algorithm, and in more complicated cases search techniques such as genetic algorithms or simulated annealing.

6 CONCLUSIONS

This brief paper gives an overview of a Battle Rhythm specification language, a work in progress, and some of its potential. Realising the full potential of such a system will require experiments with generating real Battle Rhythms. Since examination of extant Battle Rhythms does not currently reveal all of the actual requirements behind them, the work will need to be progressed with experienced ADF officers.

Further work needs to be done to develop a "compiler" for these specifications that produces meaningful error messages about deficiencies in the specification itself. The system needs to produce useful feedback about the schedules produced including how well they are using resources and how fragile they are in terms or resource dependencies. Only once the adequecy of the specification language is verified, along with the utility of the system, can we sensibly tackle questions of the best user interface.

ACKNOWLEDGEMENT

The author wishes to thank Sharon Boswell and Tim Pattison for their helpful feedback on earlier versions of this work.

REFERENCES

Duffy, L., A. Bordetsky, E. Bach, R. Blazevich, and C. Oros (2004). A Model of Tactical Battle Rhythm. In *Proceedings of 2004 Command and Control Research and Technology Symposium*.

Liu, C. L. and J. W. Layland (1973). Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM 20*(1), 46.

MacLennan, B. J. (1983). *Principles of Programming Languages: Design, Evaluation and Implementation*. Japan: Holt-Saunders International Editions. ISBN: 4-8337-0158-8.

Menzel, C. and R. J. Mayer (1998). The IDEF family of languages. In *Handbook on architectures of information systems*, pp. 209–241. Springer.

US Joint Task Force Headquarters (2012). Joint publication 3-33. Technical Report JP 3-33, Joint Task Force Headquarters.